GP Emulation and Calibration of Coupling Constants

J. Coleman, S Cao, S. Bass

May 19, 2017

Overview

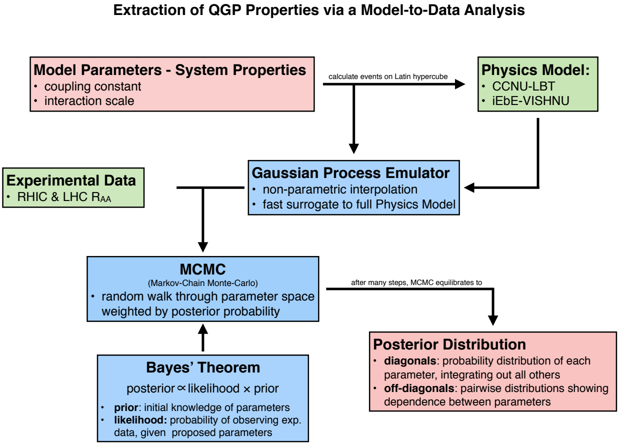
GP Review

Current Application

Prediction and Calibration

Overview

The Roll of GPs



What are GPs?

- Formally, a Gaussian Process or GP is a stochastic process Y indexed by $\mathbf{x} \in \mathcal{X}$ such that realizations are jointly Multivariate Normal.

What are GPs?

- Formally, a Gaussian Process or GP is a stochastic process Y indexed by $\mathbf{x} \in \mathcal{X}$ such that realizations are jointly Multivariate Normal.
- Practically, a GP provides a way to (very quickly!) predict an unknown function's value at new points conditional on the function's value at training points.

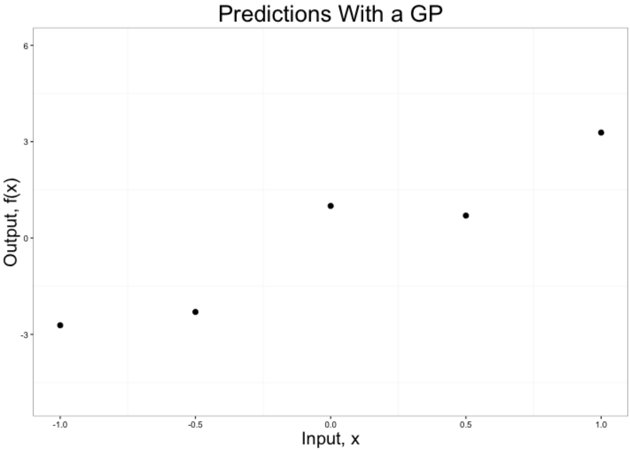
What are GPs?

- Formally, a Gaussian Process or GP is a stochastic process Y indexed by $\mathbf{x} \in \mathcal{X}$ such that realizations are jointly Multivariate Normal.
- Practically, a GP provides a way to (very quickly!) predict an unknown function's value at new points conditional on the function's value at training points.
- A GP says, essentially, if the inputs are close then the outputs should be close
- It is completely determined by a **mean function** $\mu(\cdot)$ and a positive-definite **covariance function** $c(\cdot, \cdot)$ through

$$\mu_i = \mu(\mathbf{x}_i) \qquad\qquad \Sigma_{ij} = c(\mathbf{x}_i, \mathbf{x}_j)$$

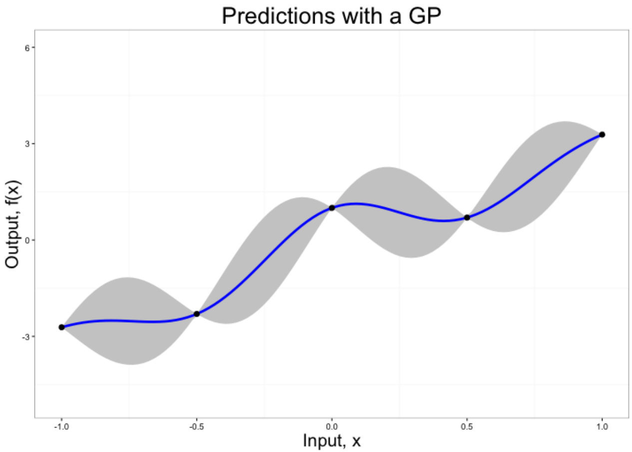
GPs In Action - Toy Example

First, the points we're trying to predict



GPs In Action - Toy Example

Prediction = mean + uncertainty



The gray bands are 95% confidence intervals.

How It Works

Let

- $f(\cdot)$ be our unknown function
- $\mathbf{x}_{\text{train}}$ be the training data
- $\mathbf{x}_{\text{new}}$ be the points we are trying to predict

$$\begin{pmatrix} f(\mathbf{x}_{\text{new}}) \\ f(\mathbf{x}_{\text{train}}) \end{pmatrix} \sim MVN\left[\begin{pmatrix} \mu(\mathbf{x}_{\text{new}}) \\ \mu(\mathbf{x}_{\text{train}}) \end{pmatrix}, \quad \begin{pmatrix} c(\mathbf{x}_{\text{new}}, \mathbf{x}_{\text{new}}) & c(\mathbf{x}_{\text{new}}, \mathbf{x}_{\text{train}}) \\ c(\mathbf{x}_{\text{train}}, \mathbf{x}_{\text{new}}) & c(\mathbf{x}_{\text{train}}, \mathbf{x}_{\text{train}}) \end{pmatrix}\right]$$

How It Works

Let

- $f(\cdot)$ be our unknown function
- $\mathbf{x}_{\text{train}}$ be the training data
- $\mathbf{x}_{\text{new}}$ be the points we are trying to predict

$$\begin{pmatrix} f(\mathbf{x}_{\text{new}}) \\ f(\mathbf{x}_{\text{train}}) \end{pmatrix} \sim MVN\left[\begin{pmatrix} \mu(\mathbf{x}_{\text{new}}) \\ \mu(\mathbf{x}_{\text{train}}) \end{pmatrix}, \quad \begin{pmatrix} c(\mathbf{x}_{\text{new}}, \mathbf{x}_{\text{new}}) & c(\mathbf{x}_{\text{new}}, \mathbf{x}_{\text{train}}) \\ c(\mathbf{x}_{\text{train}}, \mathbf{x}_{\text{new}}) & c(\mathbf{x}_{\text{train}}, \mathbf{x}_{\text{train}}) \end{pmatrix}\right]$$

$p(f(\mathbf{x}_{\text{new}}) \mid f(\mathbf{x}_{\text{train}}))$ very fast to find - Conditional Normal theory

Overview

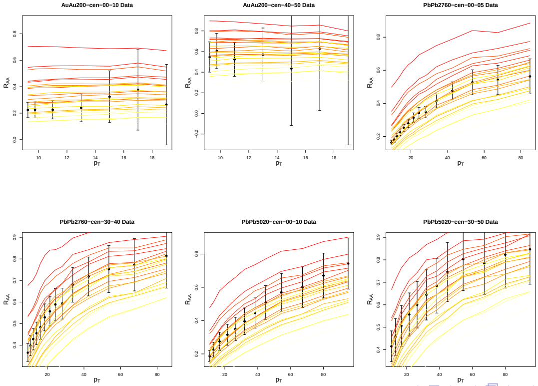
GP Review

Current Application

Prediction and Calibration

Data Exploration/setup

- Two computer input parameters: $\mathbf{x} = \{\Lambda^{\text{jet}}, \alpha_s^{\text{med}}\}$
- 3 collision systems/beach energies, each at two centralities; 6 independent datasets
- For each dataset, R_{AA} measured at 7-14 p_T values; 66 total values



Model Setup

- Let $\vec{y}^F$ be the field/experimental data
- Let $\vec{y}^R$ be the "real" values; i.e., $\vec{y}^F = \vec{y}^R$ with error
- Let $\vec{y}^M$ be the computer model data, a function of input parameter we care about
- Let Σ_y be the experimental covariance matrix

$$\vec{y}^F = \vec{y}^R(\mathbf{x}^*) + \epsilon \tag{1}$$

$$\vec{y}^R = \vec{y}^M(\mathbf{x}) \tag{2}$$

$$\epsilon \sim \mathrm{N}(0, \Sigma_y) \tag{3}$$

Analysis Steps

$$\vec{y}^F = \vec{y}^R(\mathbf{x}^*) + \epsilon \tag{4}$$

$$\vec{y}^R = \vec{y}^M(\mathbf{x}) \tag{5}$$

$$\epsilon \sim \mathrm{N}(0, \Sigma_y) \tag{6}$$

Analysis Steps

$$\vec{y}^F = \vec{y}^R(\mathbf{x}^*) + \epsilon \tag{4}$$

$$\vec{y}^R = \vec{y}^M(\mathbf{x}) \tag{5}$$

$$\epsilon \sim \mathrm{N}(0, \Sigma_y) \tag{6}$$

1. Train GPs on $\vec{y}^M$ so that for any $\mathbf{x}$ we can quickly predict $\vec{y}^M$

Analysis Steps

$$\vec{y}^F = \vec{y}^R(\mathbf{x}^*) + \epsilon \quad (4)$$

$$\vec{y}^R = \vec{y}^M(\mathbf{x}) \quad (5)$$

$$\epsilon \sim \mathrm{N}(0, \Sigma_y) \quad (6)$$

1. Train GPs on $\vec{y}^M$ so that for any $\mathbf{x}$ we can quickly predict $\vec{y}^M$
2. Perform inference on $\mathbf{x}^*$
   2.1 Propose value $\mathbf{x}^*_{\text{prop}}$
   2.2 Predict $\vec{y}^M(\mathbf{x}^*_{\text{prop}})$
   2.3 Accept based on likelihood of $\vec{y}^F$

Experimental Structure

- To simultaneously calibrate on all 6 datasets, we concatenate all experimental data
- I.e., $\vec{y}^F, \vec{y}^M \in \mathbb{R}^{66}$

Experimental Structure

- To simultaneously calibrate on all 6 datasets, we concatenate all experimental data
- I.e., $\vec{y}^F, \vec{y}^M \in \mathbb{R}^{66}$

$\Sigma_y \in \mathbb{R}^{66\times 66}$ is important. We treat as 6-block matrix; for kth block Σ_y^k:

$$\Sigma_y^k = \Sigma_{\text{sys}}^k + \Sigma_{\text{stat}}^k \tag{7}$$

$$\Sigma_{\text{stat}}^k = \sigma_{i,k}^{\text{stat}}\sigma_{j,k}^{\text{stat}}\delta_{ij} \tag{8}$$

$$\Sigma_{\text{sys}}^k = \sigma_{i,k}^{\text{sys}}\sigma_{j,k}^{\text{sys}}\exp\left[-\left(\frac{p_{i,k}-p_{j,k}}{\ell_k}\right)^{\alpha}\right] \tag{9}$$

- $p_{i,k}$ is the ith p_T value of dataset k.
- $\boldsymbol{\sigma}_k^{\text{stat}}$ and $\boldsymbol{\sigma}_k^{\text{sys}}$ are the statistical and systematic errors, respectively, associated with R_{AA} values of collision system k

Overview

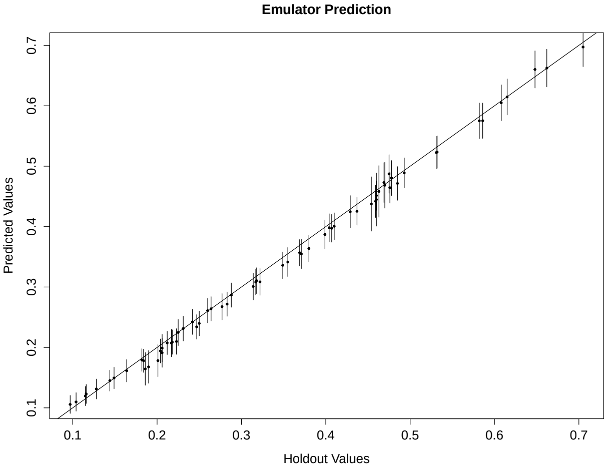
GP Review

Current Application

Prediction and Calibration

Validating the Emulator

To make sure our emulator predicts the computer model well, I trained it on 23 computer runs and predicted a holdout set.

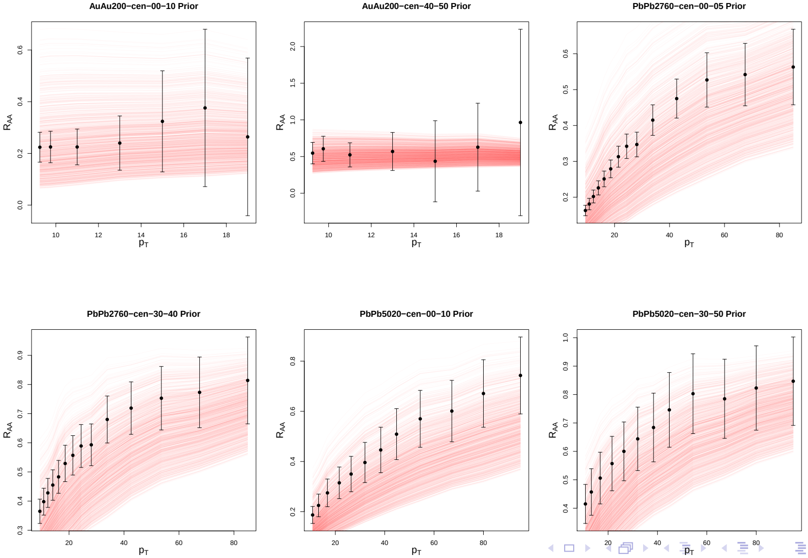


Analysis Steps

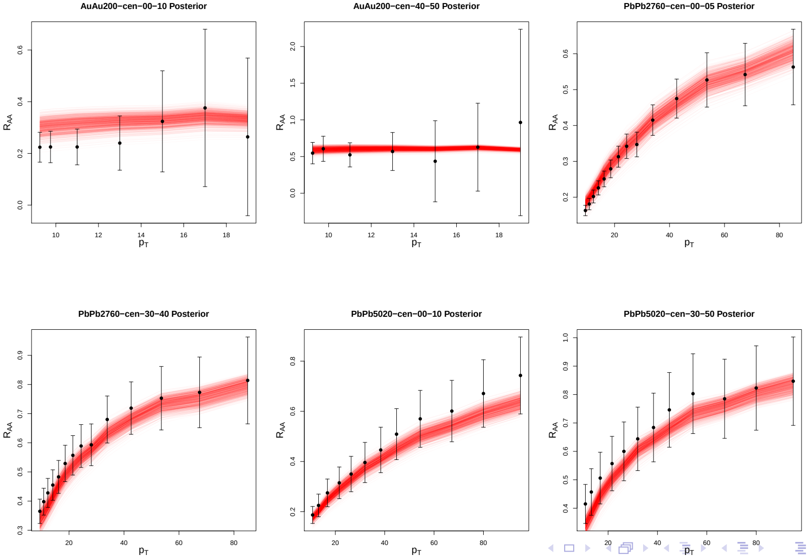
$$\vec{y}^F = \vec{y}^R(\mathbf{x}^*) + \epsilon \tag{10}$$
$$\vec{y}^R = \vec{y}^M(\mathbf{x}) \tag{11}$$
$$\epsilon \sim \mathrm{N}(0, \Sigma_y) \tag{12}$$

1. Train GPs on $\vec{y}^M$ so that for any $\mathbf{x}$ we can quickly predict $\vec{y}^M$
2. Perform inference on $\mathbf{x}^*$
   2.1 Propose value $\mathbf{x}^*_{\text{prop}}$
   2.2 Predict $\vec{y}^M(\mathbf{x}^*_{\text{prop}})$
   2.3 Accept based on likelihood of $\vec{y}^F$

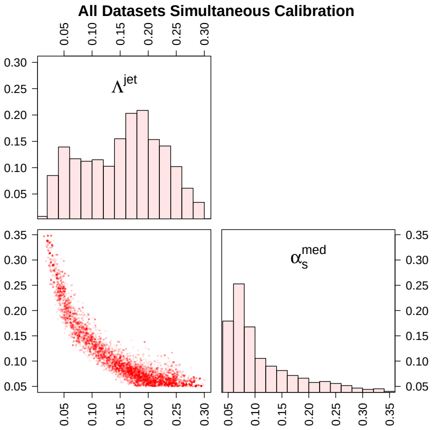
GP Predictions from Input Prior Draws



GP Predictions From Input Posterior Draws



Input Posterior Distributions



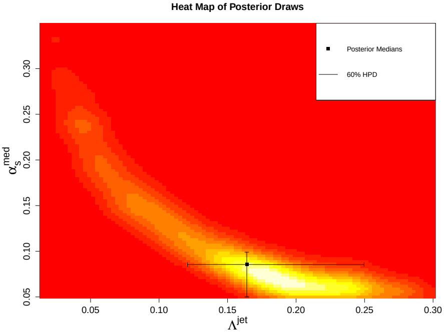


Figure: Predictive means from posterior draws of input parameters